الجزء الرابع

أساسيات وتطبيقات لغة سي

د. عمر زرتي
جامعة الفاتح
طرابلس – ليبيا

معلومات عن المؤلف

الدكتور **عمر زرتي** omarzarty@yahoo.com هو استاذ شرف بقسم الحاسب الالي بكليه العلوم جامعه الفاتح بطرابلس ليبيا.

تحصل على شهادة البكالوريوس سنة 1970 في الرياضيات من جامعة Colorado School of Mines الواقعه في مدينه Golden من ولايه كولورادو بالولايات المتحدة. تم نال شهادة الماجستير سنه 1972 من جامعة Case Western Reserve الكائنه بمدينه Cleveland بولايه اوهايو . تم تحصل على شهادة الدكتوراة سنه 1976 من جامعة Washington State University بمدينه Pullman بولايه واشنطن.

عند رجوعه الى جامعه الفاتح قام بتاسيس قسم الحاسب الالي وتراس القسم لمدة خمس سنوات، واستمر بعده كعضو هيئه تدريس في مجال (الطرق العدديه والبرمجه) ابتداء من درجة (محاضر) وانتهاء إلى درجة (استاذ).

وقد اتجه إلى مجال تاليف وكتابة الكتب المنهجيه في المستوى الجامعي والتانوي. ومن اهم كتبه كتاب (الطرق العدديه بلغة فورتران) ، كما يعتبر كتابه (اساسيات الحاسوب والبرمجه) من الكتب التي لاقت استحسانا وقبولا واسعا. إضافه إلى دلك فقد قام بتاليف عدة كتب منها (البرمجه بلغه باسكال) و (البرمجه بلغه فورتران) و(اساسيات وتطبيقات لغه سي) ، وتمت طباعه هده الكتب عدة مرات.

وللدكتور عمر زرتي ايضا اهتمام بموضوع (بحوت العمليات) الدي يقوم بتدريسه في مستوى الدراسات العليا، ويقوم بالاشراف على رساله ماجستير في هدا المجال.

الباب السابع

7

اختبـــار البيانات

Data Verification

7.1 مقدمه

في هدا الباب ، ندرس انواع البيانات وطرق اختبارها في لغة سي بمزيد من التفصيل عما درسناه حتى الان . ويمكن تقسيم هده الانواع إلى الاتي :

1 . النوع الصحيح دو الإشارة (signed int) الدي يمكن ان يكون سالبا او موجبا او صفرا .
2 . النوع الصحيح بدون إشارة (unsigned int) الدي يمكن ان يكون موجبا او صفرا فقط.
3 . النوع الصحيح الطويل long .
4 . النوع الصحيح القصير short .
5 . النوع char وهو ايضا صحيح ولكن يتكون من بايت واحدة فقط .
6 . النوع الكسري (العائم) float دو النقطه العائمه .
7 . النوع المضاعف double وله ضعف دقة النوع العائم .

ويلاحظ ان الرموز characters في لغة سي يشار إليها بارقامها في جدول ASCII (او حسب الشفرة المستخدمه لتمتيل الرموز) . وبالتالي فإن النوع النضيد string ما هو إلا مصفوفه من الارقام تناظر مصفوفه ارقام الرموز في هدا النضيد بجدول الشفرة المستخدمه لتمتيل الرموز .

7.2 النوع char

إذا كان المتغير x من النوع الصحيح بحيث لا يتجاوز

$$\{ -128 \le x \le 127 \}$$

فقد يكون من الانسب تخصيص بايت واحدة لهذا المتغير ، فهي كافيه لتخـــزين اي عدد x في النطاق المذكور، وذلك لان اكبر رقم يمكن تمثيله بالإشارة فـــي بايت واحدة هو

$$2^7 -1$$

اي 127 في النظام العشري .

وبالتالي تمكننا لغة سي من إعلان هذا النوع من المتغيرات الـــذي كثيـــرا مـــا نستخدمه في التطبيقات الإدارية مثل :

1 . المتغير الذي يعبر عن احد الاختيارات في القائمه الرئيسيه.
2 . المتغير الذي يعبر عن رقم صفحة في تقرير صغير.
3 . المتغير الذي يرمز لعدد الطلبة في فصل دراسي صغير .

ولكن قد يلاحظ الدارس انه لم تتم في جميع هذه الامثله الاستفادة مـــن الاعـــداد السالبه، وان الحد الاعلى هو 127 ، اي اننا لم نستفد إلا من 127 رقـــم. فـي الحقيقة ، يمكننا حل هذه المشكلة في لغة سي باستخدام معدل النوع:

unsigned

(اي بدون إشارة) . اي باستطاعتنا تعديل النوع char (او اي نــوع اخــر) بواسطه معدل النوع unsigned على النحو التالي :

unsigned char x ;

بهده الطريفه يصبح النطاق للمتغير x هو :

$$0 \leq x < 255$$

وهو يحتوي على عدد من الارقام يساوى

$$2^8 - 1 = 255$$

اي عدد الارقام التي يمكن تمتيلها في 8 خانات تنائيه (بايت واحدة). والسبب في دلك ان خانه الإشارة اضيفت إلى العدد واصبح لدينا 8 خانات تنائيه (بايت واحدة) بدلا من 7 خانات تنائية في حالة وجود خانة للإشارة .

مثل : مادا يطبع البرنامج التالي ؟ ومادا يحدث إدا تم تغيير الشرط x < 127 بالشرط x < = 127

```
main()
{
        char x;
        for(x=-127 ; x<127; x++)
                printf(" %d ",x);
}
```

الشكل (7.2.1) النوع char

يطبع البرنامج الارقام من 127- إلى 126 بزيادة 1 في كل مرة. وهو في داخل النطاق المسموح به للمتغير x من النوع char .

لاحظ ان عدم تحديد الإشارة signed او unsigned يعنى في لغة سي انه من النوع الاول اي singned .

إذا تم تغيير الشرط

x < 127

بالشرط

x < = 127

فمعنى ذلك ان الخروج من الحلقة يتطلب وصول x إلى القيمة 128 وهذا لــن يتحقق لان الحد الاقصى للنوع char هو 127 مما يؤدى إلى حلقة لانهائيـــه .

لاحظ ان جملة التعيين

x = 128 ;

يقبلها مصرّف لغة سي حتى لو كانت x من النوع char ، ولا يصدر رســالة خطا error massage كما نتوقع ، بل يعتبرها

x = -128 ;

اي ان إضافـة 1 إلى اكبر قيمة لها تاثير البداية من اول قيمـــة فـــي النطـــاق المحدد .

مثل (7.2.2) : اعد كتابة الـبرنامج (7.2.1) مع تحـ ـديد x مــن النــوع unsigned char، اي بـدون إشارة، ودلك لطباعة الاعداد من 0 إلى 254 .

يبين الشكل (7.2.2) البرنامج المطلوب ، وفيه نلاحظ ما يلي :

1 . عدم تجاوز x النطاق المسموح به للنوع unsigned char ، حيث تتوقف دورة for عندما تصبح قيمة x هي 255 .
2 . إدا استخدمنا النضيد " %c " بدلا من النضيد " %d " في طباعة عدد من النوع char سنحصل على الرمز المقابل لهذا العدد في جدول اسكى .

```
main()
{
        unsigned char x;
        for(x=0;x<=255;x++)
                printf(" %d ",x);
}
```

الشكل (7.2.2) النوع unsigned char

فمثلا إدا كان المتغير x من النوع char فإن جملة التعيين

x = 65 ;

تكافئ الجملة

x = 'A'   ;

مثال (7.2.3) : اكتب برنامجا لطباعة جميع الرموز في جدول اسكى بمعـــدل 10 رموز في كل سطر بحيث يطبع الرمز إلى جانب رقمه في الجدول .

```
main()
{       unsigned char k;
        printf("\n ");
        for(k=32; k < 255 ; k++)
        {       if(k%10==0)
                {       printf("\n\n");
                        getch();
                }
                printf("%3d %c  ",k,k);
        }
        printf("%3d %c",255,255);
}
```

الشكل (7.2.3) برنامج جدول اسكى

لاحظ ان الرموز الـقابلة للطباعة في جدول اسـكى تبدا من الــرقم 32 ، وان اخر رمز قد تمت طباعته في البرنامج كان خارج حلقة for (لماذا ؟) .

7.3 معدلات النوع short و long

يستخدم معدل النوع short لتعديل النوع الصحيح ليصبح ذا سعة 2 بايت (اي 16 خانة ثنائية) ، اي انه يكافئ في معظم الحاسبات النوع int ، حيث الحد الاعلى لهذا النوع هو العدد

$$2^{15} - 1 = 32767$$

والحد الادنى هو 32768 - .

إذا كانت هذه السعة لا تكفى لتخزين العدد (مثلا في تعداد السكان) يمكننا زيادة سعه الكلمه إلى 4 بايت (اي 32 بت) باستخدام المعدل long . اي ان العدد في هذا النوع يمكن ان يصل إلى

$$2^{31} - 1 = 2147483647$$

ملاحظات :

1 . عند قراءة أو طباعة عدد من النوع الصحيح الطويل long int نستخدم النضيد " % ld " وليس " %d " .

2 . عند قراءة أو طباعة عدد من النوع الصحيح غير السالب unsigned int تستخدم النضيد " %u " .

مثال : البرنامج التالي يوضح طريقة إعلان وطباعة متغير صحيح من النوع الطويل long ومتغير من النوع غير السالب .

```
main()
{       unsigned  int x;
        long y;
```

```
        x=65000;
        y=2134567890;
        printf("\n %u ", x);
        printf("\n %ld",y);
}
```

الشكل (7.3.1) طباعة متغير من النوع long والنوع unsigned

7.4 النوع المضاعف double

سبق وان درسنا النوع float ، وهو من النوع المستخدم عادة في تمثيل الاعداد الكسرية، حيث نقسم الكلمة ذات السعة 4 بايت (32 بت) إلى جزءين : الجــزء الكسرى والاس . ولكن هذا التقسيم قد يؤدي إلى حيز غير كاف لتمثيــل كــل الاعداد في الجزء الكسري.

للحصول على دقة اكثر من النوع float يمكن للمبرمج بلغة سي استخدام النوع المضاعف double الذي يوفر سعة 8 بايت .

وطبقا لمرشد توربو سي فإن نطاق العدد من النوع double يصل إلى

1.7E+308

اي ان الاس يصل إلى 308 وهو رقم كبير جدا. اما اصغر رقم فهو 1.7 E - 308 وهو رقم صغير جدا يكاد يكون صفرا .

وإذا ظلت هناك حاجة إلى ارقام اكبر يمكن استخدام النوع long double الذي يسع 10 بايت ، وهذا النوع لا يحتاجه إلا بعض علماء الفلك او الذرة وغيرها من التخصصات الدقيقة .

ملاحظات :

1 . لاحظ أن استخدام النوع double يعطى نتائج أكثر دقة من النوع float ولكن على حساب سرعة التنفيذ.

2 . عند قراءة أو طباعة عدد من النوع double استخدم النضيد "% f".

3 . عند قراءة أو طباعة عدد من النوع long double استخدم النضيد "% Lf".

مثال : البرنامج بالشكل (7.4.1) يبين الفرق في دقة النوعين float و double .

```
main()
{
	long double x;
	float y;
	x=1234567.8901234;
	y=x;
	printf("\n %Lf  %f",x,y);
}
```

الشكل (7.4.1) مقارنة بين النوعين float و double

عند تنفيذ هذا البرنامج نحصل على

1234567.875000 1234567.890123

حيث نلاحظ ان العدد الواقع على اليسار لا يوجد به خطا ، اي انه هو نفس العدد المدخل ، اما العدد الاخر ففيه خطا واضح ، حيث هناك فرق بين العدد الاصلي والمطبوع .

والسبب في ذلك طبعا هو ان الاول من النوع double والثاني من النوع float ، وكما نعلم فإن النوع double ذو سعة اكبر من النوع float.

7.5 التحويل من نوع إلى اخر

إذا كانت x من النوع float وكانت y من النوع double واستخدمنا الجملة

y = x ;

فإن ذلك يؤدى إلى عملية تحويل من النوع float إلى النوع double، وهي عملية لا ينتج عنها فقدان لاي اعداد لانها تمت من النوع الصغير (4 بايت) إلى النوع الاكبر (8 بايت) . ولكن إذا كانت العملية عكس ذلك ، اي

x = y ;

حيث تتم عملية التحويل من double إلى النوع float فينتج عنها فقد لــبعض الاعداد، لان حيز النوع float لا يكفى للنوع double، ولابد من قطع العدد y بما يكفى لسعة x .

مادا يحدث في هده الحاله عند تنفيد عبارة متل x + y ؟

هل يكون الناتج من النوع float او double ؟

الفاعدة العامه هي :

عند وجود نوعين أو أكثر فى عبارة حسابية يكون الناتج من النوع الأكبر

وعلى دلك فإن ناتج x + y سيكون من النوع double .

امـا إدا كانت x و y من نفس النوع فيكون الناتج طبعا من النوع نفسه . مثل :

6 / 5 = 1

حيث نفقد الجزء الكسري لان 6 و5 من النوع الصحيح ، اما إدا كان احـــدهما كسريا والاخر صحيحا فيكون الناتج كسريا ، متل :

6.0 / 5 = 1.2

لدلك إدا كانت k و m النوع الصحيح int واردنا ان يكون ناتج قسمة k علـى m من النوع الكسري ، يمكننا كتابة القسمة على النحو التالي :

(float) k / (float) m

او ببساطه اكتر:

(float) k / m

اي ان k (float) هي عملية تحويل من النوع الصحيح إلى الكسرى float .

مثال (7.5.1) : مادا يطبع البرنامج التالي ؟

```
main()
{
     int k=6,  m=5;
     float x;
     double y=1/3.  ;
     x=k/m;
     printf("\n %f", x);
     printf("\n %f",x+y);
     printf("\n %f", (float)k/m);
}
```

الشكل (7.5.1)

ناتج التنفيد هو

1.000000
1.333333
1.200000

7.6 اختبار البيانات

من مشاكل استخدام الحاسوب في مختلف التطبيقات مشكلة إدخال بيانات خاطئة للحاسوب ، وهو خطا بشري لا نملك تفاديه بالكامل، و لكن يمكننا التقليل منه قدر الإمكان ، وذلك بالتاكد مما يلي :

1 . القيمة المدخلة تقع في النطاق المتوقع .
2 . القيمة المدخلة من النوع المطابق للمطلوب .

ويمكن للمبرمج بلغة سي ان يستعين بعدد من الدوال الجاهزة المعرفة في ملف ctype.h التي تساعد في التحقق من نوع البيانات كما هو مبين بالجدول (7.6.1).

<u>مثل</u> اكتب برنامجا لقراءة اسم بحيث

1 . لا يزيد عدد حروفه عن 20 حرفا .
2 . لا يحتوى إلا على حروف او فراغات .

اي ان البرنامج لا يقبل إدخال غير الحروف والفراغات ، و إذا خالف مشغل البرنامج ذلك ، يصدر إنذار ولا يقبل الرمز المدخل .

الدالة	الشرط (لتكون True)
isalnum(c)	c حرف او رقم (من 0 إلى 9)
isalpha(c)	c حرف
isdigit(c)	c رقم (من 0 إلى 9)
iscntrl(c)	c رمز تحكم او إلغاء delete
isascii(c)	c احد رموز جدول اسكى
isprint(c)	c رمز قابل للطباعة
isgraph(c)	تكافئ isprint باستثناء الفراغ
islower(c)	c حرف صغير Lower case
isupper(c)	c حرف كبير Capital
isspace(c)	c فــراغ او tab او carraige return او newline او tab vertical او form feed
isxdigit(c)	c رقم سادس عشر (اي 0 , 1 , 2 , ... , 9 , A , B , C , D , E , F)

جدول (7.6.1) دوال لاختبار البيانات

```
#include <stdio.h>
#include <ctype.h>
#define N 20
main()
{
	char name[N];
	int i;
	printf("\n Please enter your name-->");
	for(i=0;i<=N-1  ;i++)
	{
		for(;;)
		{	name[i]=getch();
			if(name[i]==13)break;
			if( isalpha(name[i]) || name[i]==' ' )
			{
				putchar(name[i]);
				break;
			}
			else
			printf("\a");
		}
		if ( name[i]==13) break;
	}
	name[i]='\0';
	printf("\n The name entered is %s \n ", name);
	getch();
}
```

الشكل (7.6.1) برنامج إدخال الاسم بعد التاكد من صحه الإدخال

ملاحظات عن البرنامج (7.6.1)

1 . ضرورة التوجيه

```
# include  < ctype.h >
```

نظرا لاستعمال الدالة isalpha( )

2 . استخدام حلقة لانهائية لقراءة كل حرف من حروف الاسم لا خروج منها إلا بقراءة حرف او فراغ.

3 . استخدام الدورة الخارجيه لقراءة رموز الاسم (الحروف والفراغات) وعددها لا يتجاوز 20 رمزا، ويمكن إنهاء الحلقة بالمفتاح enter (رقمه 13 فـي جدول اسكى).

4 . لإصدار جرس التنبيه للخطا نستخدم

```
printf( " \a " )  ;
```

5 . وضع رمز الإنهاء ' 0\ ' كاخر رمز في النضيد كما هو متعارف عليه في لغة سي.

6 . اخر جملة في البرنامج

```
getch( ) ;
```

هـ-ي فقط لغرض الإبقاء على شاشة الإخراج إلى حين إدخال اي رمز .

7.7 تمارين

1 . مادا يطبع البرنامج التالي

```
main( )    ;
{   char    k   ;
for (k = 1 ; k < 200 ,k++)
printf( " %d " , k )  ;
}
```

2 . اكتب البرنامج الذي يطبع الحروف الابجدية

a b c d .... x y z

وذلك بالاستعانة بارقام هذه الحروف في جدول اسكى .

3 . ما هو الفرق بيم النوع char والنوع unsigned char ؟

4 . اكتب برنامجا لحساب !n حيث

$$n! = 1 * 2 * 3 * .... * (n-1) * n$$

ونظرا لان الناتج قد يكون اكبر من سعة النوع الصحيح int ، خاصة إذا كان n عددا كبيرا ، فمن الضروري في هذه الحالة استخدام المعدل long . ما هي القيمة n التي يبدا عندها هذا الشرط ضروريا ؟ اختبر برنامجك ببعض القيم حتى تجد القيمة الحرجة.

5 . اكتب برنامجا لحساب المجموع

$$sum = 0.001 + 0.001 + ... + 0.001$$

وذلك باستخدام النوع float مرة ثم النوع double مرة اخرى . احسب المجموع لعدد 1000 حد . كم يجب ان يكون الناتج ؟ وماهي القيمة التي يطبعها البرنامج في الحالتين ؟ كيف تفسر النتيجة ؟

6 . يريد مبرمج ان يعد برنامجا للحسابات الجارية في مصرف ، حيث قد يصل رصيد الزبون إلى ملايين الدنانير . ما نوع المتغير الذي يجب ان يستخدمه لهذا الرصيد؟ وإذا كان عدد الزبائن قد يصل إلى مئات الالاف ، ما هو نوع المتغير الذي قد يستخدمه لرقم الزبون المتسلسل؟

7 . إدا كان

i من النوع الصحيح int

n من النوع الصحيح long int

x من النوع العائم float

y من النوع المضاعف double

ما هو نوع الناتج للعبارات التاليه :

(a) i + n
(b) i / n
(c) x + n
(d) x* y
(e) n *y
(f) (float) y
(g) (double) y
(r) (float) i / n

8 . ما هي الانواع المناسبة لاختبار البيانات التالية :

(a) رقم الطالب (ويتكون من 5 خانات رقميه)

(b) التاريخ على الصورة : اليوم / الشهر / السنه

حيث اليوم : من 1 إلى 31

الشهر : من 1 إلى 12

السنة : من 1 إلى 9999

(c) الاسم : ويتكون من 20 حرفا (بما في ذلك الفراغات) .

9 . اكتب برنامجا يقوم بقراءة نضيد يتكون من حروف وارقام بحيث :

(1) لا يزيد عدد رموزه عن 15 رمزا .

(2) لا يحتوى إلا على حروف وارقام وفراغات .

وبحيث لا يقبل البرنامج اي رمز اخر ويصدر جرس إنذار بذلك .

10 . اكتب برنامجا يقوم بقراءة نضيد ثم يقوم بتحويل جميع الحروف الصغيرة (a ,b ,c ,...) إلى حروف كبيرة (A ,B ,C ,...) ويطبع النضيد.

ملاحظة : استخدم الدالة islower واستعن بجدول اسكى في عملية التحويل من حرف صغير إلى حرف كبير